Theme Division and Team Activities Interactive Teaching Method for Software Engineering

Yi Yang*, Dekuang Yu
School of Biomedical Engineering, Southern Medical University, Guangzhou, Guangdong Province, China

* Corresponding author. Tel.: 13660042367; email: yiyang20110130@163.com
Manuscript submitted January 24, 2019; accepted March 8, 2019.
doi: 10.17706/jsw.14.7.340-349

Abstract: With abstract content and complex process relationship in software engineering course, under the traditional classroom teaching method, students find it hard to grasp the core of this subject, let alone apply it in actual software projects. In order to improve the learning outcomes of software engineering, we proposed the theme division and team activity interactive teaching method by which students can truly understand and use the tools, processes and methods of modern software engineering. Based on the content attributes of the software life cycle and each stage, the themes are extracted and constructed. With instructor's guidance, students carry out team activities, including team form-up, tasks collection, group learning, topic reporting, defense review, and team feedback. The interactive teaching ways between the instructors and the learners promote the latter to actively participate in problem research and discussion, use software engineering principles and methods to solve design, development and management problems in software engineering, and enhance their technique ability and teamwork awareness.

Key words: Interactive teaching, software engineering, team activities, theme division.

1. Introduction

The teaching goal of software engineering is to cultivate the basic knowledge-based and practical capability of the software-majored in software design and development, which focuses on the application of engineering techniques and management methods: on the one hand, namely the development and improvement of software analysis, design, construction, implementation and maintenance capabilities from the view of software development methods and technologies, and on the other hand, the cultivation and improvement of engineering ability from the aspect of software engineering organization and management [1]. The course not only involves the knowledge of computer science, but also integrates relevant knowledge of mathematics, management science, engineering, sociology, and other disciplines. As the content being rich, the process relationship complex, and the principle abstract, under the traditional classroom teaching method, students find it rather hard to grasp the core of this subject, and gradually lose interest in further study. It turns out to be difficult for them to grasp its essence and apply the theory in the application. In order to better develop software engineering teaching, we proposed the theme division team activity teaching method, for students to understand and use the methods, processes and tools of modern software engineering.

Theme Division is based on the teaching purpose and key content. The teaching content is divided into a set of themes according to their attributes as well as relationships, and an in-depth analysis and explanation of each theme is carried out. It is suitable for courses with complex content and abundant knowledge.

Compared with the traditional flat model of teaching depended on the order of the chapter system, it is strongly featured as the in-depth model teaching of the intensive model [2]. Theme teaching mode requires that each theme's teaching objectives are clear, teaching content specific, and teaching design accurate, and according to the characteristics of different teaching content, diversified teaching approaches are adopted to achieve the results of excellence.

2. Theme Division Team Activity Teaching Method for Software Engineering

2.1. Theme Division and Design

The software life cycle, also known as the software system development life cycle, is the life cycle of software production from the start till scrapped. There are stages of problem definition, feasibility analysis, overall description, system design, coding, debugging and testing, acceptance and operation, and maintenance upgrade to waste in the cycle. This time-divided approach is a principle of thought in software engineering [3]. That is, step by step, each stage must be gone through - defined, work, review, document for communication, and review to improve the quality of the software. However, with the maturity of new object-oriented design methods and technologies, the gaps between the life cycle stages are tend to be blurred in some way but never mixed up in essence. Each stage of the life cycle has a defined task and produces a certain specification of documentation (information) that is submitted to the next cycle as a basis for continued work. According to the software life cycle, software development no longer only emphasizes "coding", but outlines the entire process of software development. Software engineering requires that the beginning of each new cycle of work must only be a continuation of the "correct" premise of the results of the previous cycle. Therefore each cycle progresses in a phase of activity-result-review-reactivity until the result is correct.

Software engineering course contents can be divided into seven themes according to the content nature of the software life cycle and each stage: software process, software demand analysis, software design, software implementation, software testing, software quality, and software management [4]. There is no limit to sequential relationship between the various subjects. Instead, there are sequential, parallel, reciprocal, iterative, and progressive relationships, as shown in Fig. 1.

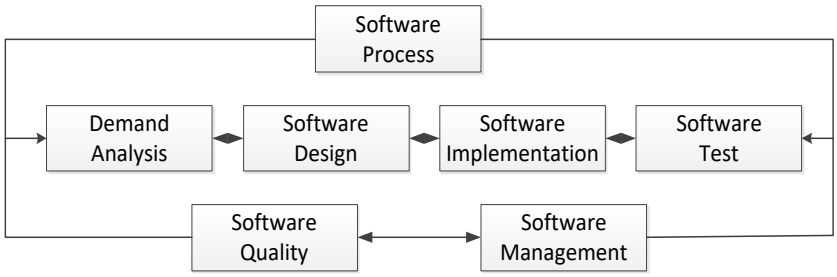

Fig. 1. Themes structure of software engineering.

Taking "software process" theme as an example, this paper introduces the teaching mode in the extraction and design of theme scheme. The software process model is a kind of development strategy which provides a set of paradigms for each stage of software engineering, so as to ensure that the progress of the project reaches the desired goal as close as possible. The development of software, with variant size, requires the selection of a suitable software process model based on the nature of the project and application, the history system, the methods used, the controls required, and the characteristics of the product to be delivered, and even the customer's habits [5]. An unsuitable choice of a certain process model will mislead direction of software development or cannot match requirements of the stakeholders. Among the many process model methods, the prototype method is one of the most effective models in practice. Compared with other models

such as waterfall model or increment model, prototype method can effectively solve the problems of low development efficiency, failure, and increased cost caused by unknown or changed user demands. It is one of the best practice methods for demand engineering, and the basis of the object-oriented analysis method. Therefore, in the practice theme of software process model selection, this method needs to be explained in detail so that students can understand thoroughly and apply it skillfully in example cases [6]. The teaching goal of this theme is set as the following: to master the principle and application of different process models, and to be familiar with the entire process of the popular process models; to understand the similarities and differences among the models; to find out the key link and matters needing attention to the application of each method; to develop the engineering practical skills in subject practice including user communication and guidance, teamwork, interpersonal technical exchange, product business analysis, demand documentation, experience sharing, problem raising and resolution, product observation, discovery and development capabilities, intelligent thinking and innovation capabilities.

2.2. Project Design Based on Themes

The project design complies with the main goals of software engineering practice training, and takes into consideration factors such as the authenticity of the project, the difficulty in understanding the requirements for students, and the combination of the project and professional courses. The selection of projects should come from or be close to real projects provided by software companies, which reflect the actual need and application of software engineering thoughts and methods [7]. In addition, students are more likely to accept projects that occur in their familiar scenes, or projects that they have the willingness to actively explore, such as e-commerce platforms, game development, and entertainment industry applications.

Combining the main goals of software engineering practice training and previous teaching experience, three types of projects are introduced for students to choose in the project design of software engineering practice training. These three types of projects basically cover the main categories of enterprise operations in the employment market, i.e. for end consumers, for industries, and for institutions. End-oriented consumers projects combine software development technologies (that students are familiar with) with applications (that consumers are interested in), and cover the unique needs of end-consumers such as "e-commerce platforms", "online payment", "online game", and "real-time positioning". All are familiar scenes for students. Industry oriented projects include intelligent agricultural monitoring, intelligent access control system, etc., which belong to the sphere of the industry fields such as agriculture, traffic, security check. Under the background of more and more developed intelligent technology services, they have a good application and development prospect. Institution-oriented projects include video and audio live broadcast and point-dial systems. They mainly aim at the development of publicity, education, and training institutions, and require students to meet up with "dynamic information interaction", "video playback", "multi-forum" requirements. All these projects aim to enhance students' ability to actively participate and use knowledge during the training process, and strengthen the practicality of software engineering..

2.3. Team Activities in Teaching

Using traditional teaching methods, most of the time students can only learn passively, and find it rather difficult to understand the kernel of software engineering. Meanwhile, in the process of software development, under the constraints of schedule, quality and cost, software development activities need to be completed in a team. However, the traditional software engineering practice task is traditionally set as a single question, which means the software design is restricted to be small in scale, and further away from practical problems and situations. Thus there has formed a fixed mode of demand design and architecture design, and it is difficult to innovate in this constraint. It turns out that it is the same person who completed all the stages, from start to end. No partners, no team interaction, and it may be found at last that the functional design is

unreasonable and incomplete, or the learners never have the chance to find potential loopholes in the project when testing. Without a team, it is difficult to ensure the correctness, integrity and applicability of the software system. From the above problems, it can be seen that the centralized practice mode has destroyed the systematic link of the software design and development industry. Moreover, it is easy for students to ignore the links between related courses and almost impossible to make full use of the knowledge they have learned at school in their future work. That's why we are determined to explore some innovative ideas of practice teaching.

The team activity teaching is a popular teaching or academic communication method in the classroom of European and American universities. Students claim to study and discuss tasks, consult relevant documents before class, make summary, write and refine reports, and in this way instructors can fully mobilize students' enthusiasm, encourage them to actively participate, and jointly discuss on research issues from multiple perspectives, in multiple directions, and of multiple levels. This will lead to a better understanding of the teaching content. The core of team activity teaching is to fully exploit the learning potential of course participants and maximize multi-angle and multi-level cognitive interaction so as to deepen the understanding of a certain topic and achieve the best results of academic exchanges.

2.4. Conduct of Team Activities

The team activity teaching in software engineering course consists of four steps.

Forming the teams. According to the specific content of the software engineering course, the students are divided into several teams. Each team consists of 5-7 people. A chairman in charge is selected to manage and coordinate the activities of the entire group. Following the principle of free association, with the "ability balance, background diversity" guidance, students are grouped through mutual personal communication.

Projects claim and tasks assigned. After the formation of the team, the students are free to select topics through discussion. Each team can select its project from the subject library given by the instructor at the beginning of the course or they can set topics according to their interests, and then they strive together to work on it during the entire course. The final decision needs to be discussed by the members in the team to reach an agreement. The team leader has the responsibility of coordinating the division of labor and balancing the workload of all members. It means that the team leader must assign the sub-tasks of the project to each member based on individual interests and preference as well as the team's target. Team members can work either independently of corporately according to the needs of tasks, to finally obtain project planning plans, demand analysis reports, software outline design reports, software detailed design reports, and software test plans. All task timelines are coordinated by the team leader.

Lectures are conducted interactively with team learning activities. Each theme consists of both class teaching and team activities. Through the former teaching forms, the theoretical knowledge system is passed on to students. Students gradually internalize knowledge through self-learning and reviewing data after class. Through group activities in the classroom and after class, students can communicate more fairly and easily between them and to the instructors, thus deepen the understanding of knowledge. In the teaching stage only the core part of the module rather than all aspects is covered; Group activities realizes the interaction of knowledge, technology and ideas, encourage the communication and brain storms among different roles. The task requires the team to work together, consult reference materials, discuss and ask questions, and determine the resolve plan. Under the coordination of the team leader, each member tries his best to devote to the selected simulation project. This process can strengthen the learners' ability to collaborate and cooperate, actively explore problems and solve problems during the seminar, as well as reading, writing and organizing ability, which will surely lay a sound foundation for the software development.

Team demonstration. It is warmly encouraged that students walk to the podium and speak out about their own program design. Each member has the opportunity to illustrate his own mission view and report to other

team members and the instructors for them to have a certain understanding of what he has achieved. The defense session is a key part of the evaluation report. As judges, the audience can raise doubts about the speaker's report. Unreasonable, incorrect and ambiguous places should be pointed out. The defense session is not only an exercise in the ability to express opinions, but also an examination of the content of the academic and technique aspects of the report. It can also stimulate the interest of all members to learn and actively identify the potential problems, to get in-depth understanding of the problem and internalization of related knowledge.

2.5. Interactive Teaching

Taking the design pattern as an example, this paper describes the application of interactive teaching of team activities in software engineering course. Software architecture and design pattern is an important part of software engineering theory system. This sub-theme requires students to provide general solutions to certain types of common problems in the software design process on the basis of mastering basic object-oriented (OO) programming ideas and programming languages. For most undergraduates, although they have some know about of OO concepts such as polymorphism, inheritance, and encapsulation, due to lack of practical experience, they often cannot fully understand and flexibly apply the intentions and evolutionary processes to applicable situations of design principles and patterns. For this reason we designed a four-step interactive teaching method to change the status quo.

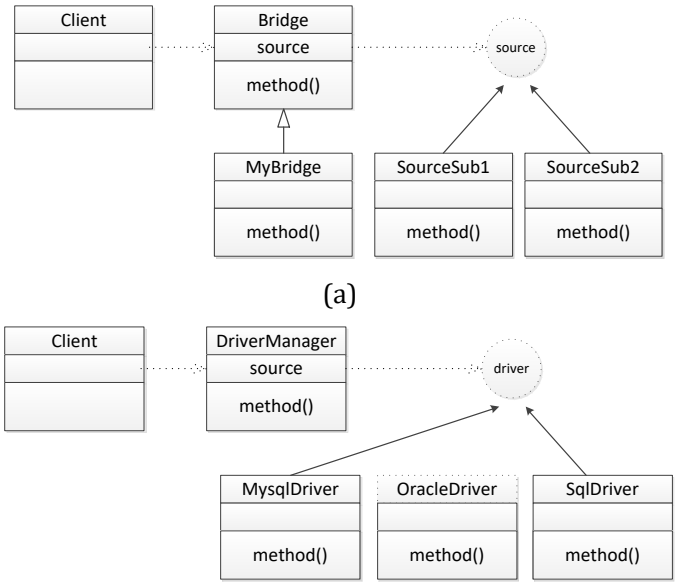

Fig. 2. Bridge pattern diagram and application.

2.5.1. Preparation

Trying to solve a coding problem completely, we use a set of cases to introduce the design pattern theme, and constantly carry out code reconstruction and design evolution, and finally give a specific model solution. Take the bridge mode for an example, the essence of which is to separate things from their specific realization so that they can change independently. The intention of the bridge is to decouple the abstraction from the implementation. JDBC Driver Manager is a typical application of bridge mode. JDBC switches among different types of databases when connecting databases as it provides a unified interface to all of them. Each database provides its own implementation. It is bridged with a program called a database driver, as shown in Fig. 2. This application requires students to master the design principles of "bridging mode" through the operation

of connecting databases.

After reading and understanding the basic requirements of database operation, students fill the relevant code according to the given design framework and complete the corresponding functions.

2.5.2. Evaluation

The instructor assesses the difficulty of pre-heating questions and the students' answers by group. Moving the evaluation for students to the beginning of the class, it can be found that most students can understand the basic design requirements of this case. Even if they have not yet mastered the design model, they can complete some of the functions through the UML means previously mastered.

2.5.3. Class discussion

At this stage, most students are familiar with the case problems raised in the above design models, and they can also find problems in the design process (such as code duplication, inability to effectively respond to changes in demand, etc.). Through cross-group discussion, the general solutions of the design mode are finally given, thus strengthening students' understanding of the complex concepts.

2.5.4. Experiment after class

Through the after-class experiment, students have the opportunity to use the design modes to finalize the programming of the case problems and testify the actual operation results. The definition of some super-classes and interfaces is given, and students are required to fill in the corresponding code.

2.6. Feedback from the Students

In the week after the course, all students who have participated in the software engineering practice training are invited to complete the feedback questionnaire. According to data collected these three years, more than 90% positive feedbacks have been received from over 100 learners on various aspects of the Theme Division and Team Activities Interactive Teaching Method. It is widely believed that this teaching mode in software engineering practice training has obviously helped to improve students' software process management and development skills, as part of the questionnaires shown below. Most students argue that this practice training mode has been a great aid for them in problem analysis and understanding, model study, development technology, process management and environment configuration, on both the front and rear sides. The percentage in the brackets is the ratio of strongly approved opinions among the investigated students.

1) Theme Division helps to clarify the structure and requirements of the whole course. (93.7%)
2) The gradual addition of software process requirements in the course is more conducive to the mastery and application of software engineering ideas. (92.4%).
3) The team interaction method has motivated me to learn, and I am able to actively participate in projects and discussions (91.3%).
4) Through the theme team activities, I have a better understanding of the meaning of requirements analysis, and mastered one or more specific demands analysis technology (92.7%).
5) Through team discussion, I have learned more about the architecture of the system and been able to analyze the architecture of the existing system and propose the appropriate architecture for the new system (87.2%).
6) I have improved my knowledge and mastery of different roles in software management by thematic team activities (91.6%).
7) With peer support, I have come to appreciate the importance of code normality and readability (90.4%).
8) I have learned more about testing tools, platforms and techniques through interactive communication and demonstration (89.4%).
9) After the course training, I prefer to work with people rather than by myself (91.4% strongly approved).

10) Through regular inter-group exchanges, I have learned a lot of new approaches from other teams (85.9).
11) I am more willing to learn skills in the way of team work and communication after this course (92.7).
12) As a result of cooperation, I hold a positive attitude towards technology demonstration as a good way to promote learning (92.8).

This paper designed a questionnaire for students' learning ability and obtains the most direct information about the learning effect. 180 questionnaires were issued, with a recovery rate and efficiency of 97%. The reliability of the questionnaire is solid, as shown in Table 1, revealing the status of students' learning ability from five aspects with Theme Division and Team Activities Interactive Teaching method . The greater the number (from 1 to 5), the higher the degree of agreement is.

Table 1. Status of Students' Learning Ability

Aspects of learning ability	5	4	3	2*c*	1
Learning motivation	76.3%	17.9%	5.8%	0	0
Learning attitudes	75.8%	22.9%	1.3%	0	0
Partner guidance	60.3%	33.9%	5.1%	0	0
Engineering learning skills	41.6%	37.3%	14.8%	6.3%	0
Learning perseverance	32.3%	53.0%	14.7%	0	0

2.6.1. Learning motivation

Learning motivation is a direct indicator of students' learning expectations, and the degree to which learning needs are met. Table 1 show that students have a strong internal motivation for learning, and have high expectations in the cognitive, interpersonal and personal fields. Among them, they hope that their comprehensive skills such as academic knowledge and interpersonal communication in the cognitive field will be improved.. The results also show that the students' expectations of learning in the personal sphere have been better stimulated, and the self-learning classroom built by teachers can better meet the students' learning needs in all aspects. Among them, the students are the most satisfied with the various activities in or outside of the classroom.

2.6.2. Learning attitudes

The students' learning attitude towards teamwork learning determines the quality of classroom learning to a certain extent. Figure 1 shows that most students have a positive attitude towards teamwork learning and ensure the quality of classroom learning. Under the positive learning attitude, students' enthusiasm and degree of participation in the project have increased to a certain extent..

2.6.3. Partner guidance

Partner guidance is the key link in the construction of college students' independent learning ability. Table 1 shows that, overall, with new teaching mode, there are frequent and higher level exchanges among members of the same group as well as different groups, with more communication in group learning and course assignments, and communication in learning plans and academic discussions be improved.

2.6.4. Engineering learning skills

Engineering learning ability is the core of software students' learning ability. It can be inferred from the data that with new teaching strategies the student's engineering learning ability development condition is overall better. Among them, the development of the individual field is the best, the development of the cognitive field is the second, and the development of the interpersonal field is the worst.

2.6.5. Learning perseverance

Anti-interference ability is a typical manifestation of learning perseverance. This questionnaire sets out a question for testing, namely, "how frequently the subject of challenge is selected in the course of study, even

though it may reduce performance." The survey found that only about 1/3 of the students chose "regular", over half of students chose "sometimes", and 14.7% students never or rarely chose challenging topics. It shows that only a small portion of students have good anti-interference ability in the learning process, and students' learning perseverance needs to be strengthened.

To sum up, teachers and students have a close relationship and good communication. Teachers are capable of playing the role of independent learning guides. Students have more clear learning motivation for themselves, learning perseverance is exercised, learning attitude is good, and engineering learning ability is improved as a whole. However, the guidance and training of students in the field of interpersonal development need to be strengthened.

3. Conclusion

The complexity brought about by the interdisciplinary, engineering and domain relevance of software engineering makes it necessary to adopt a gradual and multi-level model for teaching. The Theme Division based on content attributes can better define the specific teaching goals and requirements. Adoption of the team activity teaching method, we made effort to effectively integrate the team activities into the different modules of the software engineering course, and combine them with the traditional teaching methods, with added software engineering stage scene features, the new approach can effectively mobilize students' learning enthusiasm, exercise their self-learning ability and innovation ability, and enhance team cooperation awareness. The new teaching mode as a good two-way communication has achieved quite approved teaching effect, worthy of further promotion and application. The following results can be achieved:

Students' ability to learn actively. Through the team way of participation of engineering projects of all types of themes in the discipline sphere, students can actively study and exchange information and ideas among all members. Almost all participants can escape from passive guidance to enjoy active problem solving, thus improving the learning effect.

Students' engineering practical ability. Through the training method of the projects, students' professional skills can be greatly improved. At the same time, the ability to solve problems during the implementation of the project is also greatly improved.

Students' engineer quality. To be a qualified engineer, the learners also need to have good team cooperation ability, communication ability, self-management ability and innovation ability. Trained by Theme Division and Team Activities Interactive Teaching model, students can clearly recognize the importance of these abilities and improve in the real development process.

Acknowledgment

Y. Yang and D. K. Yu thank for the support of the key platform and research project of the Guangdong Provincial Department of Education(2016 GXJK021), the Guangdong Provincial Science and Technology Academic Monograph Project(2017 A0304009), and the Ministry of Education's Cooperative Education Project(201702071040, 201702071180, 201702085011), Guangzhou University Innovation and Entrepreneurship Education Project Curriculum and Teaching Research Project(201709k50), Guangzhou University Innovation and Entrepreneurship Education Project Feature Activity Project(201709T40), Southern Medical University School Higher Education Teaching Research(2017 JG13), Southern Medical University Mixed Teaching Project(B1040889) , Southern Medical University Student Innovation and Entrepreneurship Training Program Project (201812121178, 2018121066), Guangdong University Student Innovation and Entrepreneurship Education Research Center Project funding(201801193059), Guangdong Provincial School Enterprise Collaboration Education Project(PROJ993686684514258944).

The authors also thank the friends and students who show their great enthusiasm in and put efforts into

teaching, learning and extra-curricular practice based on Outcome Directed SPOC Teaching Model. They are BinZhang, Fa-we He, Yu-qing Wang, Ze-chen Li, Zi-jing Zhang, Wen Huang, Ying Zhou, Mu-zi Shi, and Yan-liang Li..

References

[1] Tian'en, S. (2016). On teaching strategies of outcome-based education. *Journal of Jilin Normal University,* Humanities & Social Science Edition, *3*, 83-88.

[2] Papadopoulos, P. M., Stamelos, I. G., & Meiszne, A. (2013). Enhancing software engineering education through open source projects: Four years of students' perspectives. *Education and Information Technologies, 18(2)*, 381-397.

[3] Cicirello, V. A. (2017). Student developed computer science educational tools as software engineering course projects. *Consortium for Computing Sciences in Colleges*, *32(3)*, 55-61.

[4] Paasivaara, M., Vanhanen, J., Heikkila, V. T., *et al*. (2017). Do high and low performing student teams use scrum differently in capstone projects, 146-149.

[5] Xiaoying, B, Shanshan. L, & Mingjie, L. (2018). Exploration on software engineering practical teaching based on agile development. *Experimental Technology and Management, 35(4)*, 6 -11.

[6] Chatley, R., & Field, T. (2017). Lean learning: Applying lean techniques to improve software engineering education, *(17)*, 117-126.

[7] Bandyopadhyay, S., & Thakur, S. S. (2016). ICT in education: Open source software and its impact on instructors and students. *International Journal of Computer Applications, 151(6)*, 19-24.

Yi Yang was born in Guangdong province, China in April, 1973. She got her bachelor's degree and master's degrees in computer software in National University of Defense Technology, Changsha, China in the year 1994 and 1997 respectively, and got the doctor's degree in pathology in Southern Medical University, Guangzhou, China, in the year 2005.

Since March 1997, she has been engaged in teaching and scientific research at the School of Biomedical Engineering of Southern Medical University. In 2005, she was appointed as associate professor. In recent years, she has published more than 20 scientific research articles in journals, publications and academic conferences on computer software, and has written 8 books of programming and software development, including JSP Web Application Design Case Tutorial (Beijing, China: Posts & Telecom Press, 2014), Android Mobile Application Development (Beijing, China: Posts & Telecom Press, 2017), and Data Structure (C++ version) (Beijing, China: Metallurgical Industry Press, 2009). In the past three years, she has finished two provincial funds of science and technology research projects and participated in and completed a number of national and provincial funds for scientific research projects. She has been engaged in medical image recognition and detection, quantitative pathology, mobile application development, information system development, intelligent information processing.

Dr. Yang is director of Guangdong Biophysical Society, Member of Chinese Biophysical Society, and Director of Chinese Society for Stereology. She has won 12 excellent first prizes for teaching, 2 first prizes for teaching competitions at university, and 2 National Second Prizes for guiding JAVA competitions.

Dekuang Yu was born in Jiangxi Province, China in April, 1972. He got his bachelor's degree and the master's degrees in biomedical engineering in the First Military Medical University, Guangzhou, China in the year 1993 and 1999 respectively, and got his doctor's degree in biomedical engineering in Southern Medical University, Guangzhou, China, in the year 2006.

Since March 1993, he has been engaged in teaching and scientific research at the School of Biomedical Engineering of Southern Medical University. In 2013, he was appointed as associate professor. In recent years, he has published more than 10 scientific research articles in journals, publications and academic conferences on biomedical engineering. He has finished one provincial fund of science and technology research projects and participated in and completed a number of national and provincial funds for scientific research projects. He has declared 3 national invention patents, and 2 software copyrights have been obtained. He has been engaged in medical image recognition and detection, electrocardiogram simulation, information system development.

Dr. Yu is Member of Chinese ECG Society. He has won 3 excellent prizes for guiding students to obtain 3 national and provincial science and technology competitions.